an object is created from a class stored in the class area 36. Step 4.2 checks to see if the class is a global class, i.e., a class all of whose instances are global or one whose instances we expect to quickly become global, whereby the object is assigned to be global and/or placed in the global heap (Step 4.7). Step 4.3 calculates the size of the object to determine whether it will fit in the thread heap. Step 4.4 performs garbage collection to free up memory. Step 4.5 places the object in the thread heap memory along with the object length in multiples of eight. Step 4.6 Uses a spare bit in the length attribute as a flag and sets it as local ('0'). The process ends at Step 4.8.--

In the claims:

1. (Amended) A method of managing memory in a multi-threaded processing environment including respective local thread stacks and heaps and a global heap, said method comprising:

creating an object in a thread heap; and

monitoring the object to determine whether the object is referenced only from a given thread stack.

2. (Amended) The method as claimed in claim 1 further comprising:

assigning a local status to the object;

changing the status of the object to global under certain conditions.

3. (Amended) The method as claimed in claim 2 further comprising deleting from a given thread heap one or more local objects when they are not accessible from a local root.

4. (Amended) The method as claimed in claim 3 where accessibility is determined by tracing from the local root.

5. (Amended) The method as claimed in claim 4 wherein the status of an object in the given thread heap is changed to global if the object is assigned to a static variable or if the object is assigned to a field in any other object.

6. (Amended) The method as claimed in claim 3 further comprising intercepting assignment operations to an object in the thread heap to assess whether the object status should be changed.

7. (Amended) The method as claimed in claim 6 wherein the multithreaded processing environment is a virtual machine.

8. (Amended) The method as claimed in claim 7 wherein the virtual machine comprises an interpreter and the write operation code in the interpreter is modified to perform the checking of assignment of the object.

Cont A2

9. (Amended) The method as claimed in claim 8 wherein the virtual machine comprises a just-in-time compiler, and wherein native compiled write operation code includes native code to perform the checking of assignment of the object.

10. (Amended) The method as claimed in claim 9 further comprising using spare capacity in the object header for a flag.

11. (Amended) The method as claimed in claim 10 further comprising using multiples of 2 or more bytes in a thread heap to store the objects whereby there is at least one spare bit in the object length variable and using the at least one spare bit as the flag.

12. (Amended) The method as claimed in claim 11 further comprising moving objects whose status is global from the thread heap to a global heap.

13. (Amended) The method as claimed in claim 12 further comprising compacting the accessible local objects in a thread heap.

14. (Amended) The method as claimed in claim 1 wherein certain objects are associated with a global status on creation.

15. (Amended) The method as claimed in claim 14 where said certain objects include Class objects, Thread objects and Runnable objects.

Cont A2

16. (Amended) The method as claimed in claim 14 further comprising a step of analysing whether an object is likely to be made global and associating such an object with a global status on creation.

17. (Amended) The method as claimed in claim 16 further comprising allocating objects assigned as global on creation to the global heap.

18. (Amended) A system for managing memory in a multi-threaded processing environment comprising:

respective local thread stacks and heaps;

a global heap;

means for creating an object in a thread heap; and

means for monitoring the object to determine whether the object is referenced only from a given thread stack.

19. (Amended) The system as claimed in claim 18 further comprising means for associating a local status with the object and means for changing the status of the object to global under certain conditions.

20. (Amended) The system as claimed in claim 19 further comprising means for deleting from the thread heap one or more local objects when they are not reachable from a local root.

21. (Amended) The system as claimed in claim 20 further comprising:

means for changing the status of an object in the thread heap to global if the object is assigned to a static variable or if the object is assigned to a field in any other object.

22. (Amended) A computer program product stored on a computer readable storage medium for, when executed on a computer, managing memory in a multi-threaded processing environment including respective local thread stacks and heaps and a global heap, said product comprising:

Cont A7

instructions for creating an object in a thread heap; and

instructions for monitoring whether the object is referenced only from a given thread stack.

23. (Amended) The product as claimed in claim 22 further comprising:

means for associating a local status with the object;

means for changing the status of the object to global under certain conditions.

24. (Amended) The product as claimed in claim 23 further comprising means for deleting from the thread heap one or more local objects when they are not a local root.

25. (Amended) The product as claimed in claim 24 where accessibility is determined by tracing from the local root.

26. (Amended) The product as claimed in claim 25 wherein the status of an object in the thread heap is changed to global if the object is assigned to a static variable or if the object is assigned to a field in any other object.